06 · 29 · 00

A

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

"EXPRESS MAIL" MAILING LABEL NUMBER EL170350261US
DATE OF DEPOSIT June 27, 2000
I HEREBY CERTIFY THAT THIS PAPER OR FEE IS BEING DEPOSITED WITH THE UNITED STATES POSTAL SERVICE "EXPRESS MAIL POST OFFICE TO ADDRESSEE" SERVICE UNDER 37 CFR 1.10 ON THE DATE INDICATED ABOVE AND IS ADDRESSED TO: BOX PATENT APPLICATION FEE, ASSISTANT COMMISSIONER FOR PATENTS, WASHINGTON, DC 20231.
TYPED NAME KARI BATEMAN
SIGNED

Date: June 27, 2000

File No. A-68807/AJT/JWC

Box PATENT APPLICATION FEE
Assistant Commissioner for Patents
Washington, DC 20231

Sir:

Transmitted herewith for filing is the patent application of Inventor(s): Amnon Meyers

For: AUTOMATED GENERATION OF TEXT ANALYSIS SYSTEMS

Enclosed are also:

- Prior Art Statement.
- X 8 Sheet of drawings, Formal __, Informal X
- An Assignment of the invention to:_____.
- X Declaration and Power of Attorney for Patent Application.
- X Verified Statement Claiming Small Entity Status
- Preliminary Amendment

	(Col. 1) NO. FILED	(Col. 2) NO. EXTRA	SMALL ENTITY RATE	SMALL ENTITY FEE	OTHER THAN SMALL ENTITY RATE	OTHER THAN SMALL ENTITY FEE
BASIC FEE				**$345**		**$690**
TOTAL CLAIMS	13 - 20	-0-	x 9 =	$0	x 18 =	$
INDEP CLAIMS	3 - 3	-0-	x 39 =	$0	x 78 =	$
MULTIPLE DEPENDENT CLAIM PRESENTED			+130 =	$	+260 =	$
If the difference in Col 1 is less than zero, enter "0" in Col. 2			TOTAL	**$345.00**	TOTAL	**$**

X Our Check No. 5860 in the amount of $345.00 is enclosed to cover the filing fee.

X The Commissioner is hereby authorized to charge any additional fees which may be required, including extension fees, or credit any overpayment to Deposit Account No. 06-1300 (Order No. A-68807/AJT/JWC). Two copies of this sheet are enclosed.

Respectfully submitted,

Maria S. Swiatek
Reg. No. 37,244

FLEHR HOHBACH TEST ALBRITTON & HERBERT LLP
Four Embarcadero Center, Suite 3400
San Francisco, California 94111-4187
Telephone: (415) 781-1989
Fax: (415) 398-3249
1012716

(1.14) 08/95

Applicant or Patentee: Amnon Meyers
Serial No.: Not yet assigned

Atty Docket No: A-61607/AVI/WC
Filed: Herewith

VERIFIED STATEMENT (DECLARATION) CLAIMING SMALL ENTITY STATUS (37 CFR 1.9(f) and 1.27(c)) - SMALL BUSINESS CONCERN

I hereby declare that I am an official of the small business concern empowered to act on behalf of the concern identified below:

NAME OF SMALL BUSINESS CONCERN: Text Analysis International, Inc.
ADDRESS OF SMALL BUSINESS CONCERN: 1604 Mariani Drive
Sunnyvale, CA 94087

I hereby declare that the above identified small business concern qualifies as a small business concern as defined in 13 CFR 121.12, and reproduced in 37 CFR 1.9(d), for purposes of paying reduced fees to the United States Patent and Trademark Office, United States Code, in that the number of employees of the concern, including those of its affiliates, does not exceed 500 persons. For purposes of this statement, (1) the number of employees of the business concern is the average over the previous fiscal year of the concern of the persons employed on a full-time, part-time or temporary basis during each of the pay periods of the fiscal year, and (2) concerns are affiliates of each other when either, directly or indirectly, one concern controls or has the power to control the other, or a third party or parties controls or has the power to control both.

I hereby declare that rights under contract or law have been conveyed to and remain with the small business concern identified above with regard to the invention entitled *Automated Generation of Text Analysis Systems* by inventors *Amnon Meyers*, described in the specification filed herewith. If the rights held by the above identified small business concern are not exclusive, each individual, concern or organization having rights in the invention is listed below* and no rights to the invention are held by any person, other than the inventor, who would not qualify as an independent inventor under 37 CFR 1.9(c) if that person made the invention, or by any concern which would not qualify as a small business concern under 37 CFR 1.9(d), or a nonprofit organization under 37 CFR 1.9(e). *NOTE: Separate verified statements are required from each named person, concern or organization having rights to the invention averring to their status as small entities. (37 CFR 1.27)

Name: _____
Address: _____
[ ] Individual [ ] Small Business Concern [ ] Nonprofit Organization

I acknowledge the duty to file, in this application or patent, notification of any change in status resulting in loss of entitlement to small entity status prior to paying, or at the time of paying, the earliest of the issue fee or any maintenance fee due after the date on which status as a small entity is no longer appropriate. (37 CFR 1.28(b)).

I declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application, any patent issuing thereon, or any patent to which this verified statement is directed.

Name/Title: Avi Meyers, CEO
Address: 1604 Mariani Drive, Sunnyvale, California 94087
SIGNATURE _____ DATE 6-26-00

1012093

AUTOMATED GENERATION OF TEXT ANALYSIS SYSTEMS

BACKGROUND OF THE INVENTION

Text analysis is an area of computer science that focuses on processing text to extract information through pattern recognition. The decade of the 1990's has seen an unprecedented explosion in work on learning methods for text analysis. Prior text analysis methods rely on unsupervised learning, where the system is responsible for teasing generalizations from texts or samples. One such system, the HASTEN system described in "SRA: Description of the SRA System as Used for MUC-6," Krupka, George R., pp. 221-235, Proceedings Sixth Message Understanding Conference (MUC-6), November 1995 (referred to herein as Krupka). Krupka teaches a system for grouping text samples supplied and labeled by users and creating data structures called e-graphs. The system in Krupka then uses a similarity metric to decide if portions of an input text are related to e-graphs that have been created. It applies these collections of e-graphs, called collectors, as sequential processing phases, in order to match each sample set to the input text. Generalization of the elements of e-graphs is performed manually by the developer. There is no notion of generating grammar rules from e-graphs. The work does not establish a method for converting the collectors to rule-based passes of a text analyzer. The work does not describe a way to automatically generate substantial portions of a text analyzer. The system in Krupka requires a large amount of user interaction to perform tasks manually beyond adding and labeling samples, and was applied specifically to create an event level pattern for MUC text analysis. However, Krupka's system does not teach a general and fully automated text analyzer capability.

Another text analysis system is disclosed in Huffman (U.S. Patents 5,796,926 and 5,841,895). The Huffman patents deal with text extraction at the event level and teach methods for locating potential event patterns of interest. In essence, Huffman teaches a rigid, inflexible method

of searching for specific patterns such as "actor acts on object."

There is a need for a system that automatically generates text analysis systems with minimal training samples while retaining sufficient intelligence to recognize patterns beyond those described by the training samples, sufficiently flexible to allow adaptation to a variety of applications.

SUMMARY OF THE INVENTION

An embodiment of the present invention includes a generator program 106 that utilizes a hierarchy of user-supplied samples and a text analyzer framework to create complete text analyzer programs. The hierarchy and framework are related in that the top-level concepts of the hierarchy are associated with stubs, or empty regions of passes, in the text analyzer framework. The generator program fills these stub regions with text analyzer passes generated from samples in the hierarchy. A user guides the conversion of the samples to generalized rules for recognizing not only the given samples, but also related patterns that are processed at a later time. Users may supply additional samples in order to process novel patterns that were not anticipated when the initial text analyzer was created. When a text analysis system according to the present invention fails to identify a pattern, a user can simply highlight the unrecognized sample in text and label its components, if necessary, to enable the generator to create a new text analyzer that now recognizes the new sample and related samples processed at a later time. Rather than using a similarity metric, an embodiment of the present invention applies rules that have been automatically generated from samples.

BRIEF DESCRIPTION OF THE DRAWINGS:

FIG. 1 is an illustration in block diagram form of various components of an embodiment of the present invention;

FIG. 2 is a flow chart illustrating the steps executed by a text analyzer program produced by the present invention;

FIG. 3 is an illustration of a parse tree data structure created and maintained by the present invention;

FIG. 4 shows a sample hierarchy constructed and used by the present invention;

FIG. 5 is a flow diagram illustrating the major steps of rule generation and rule merging according to methods of the present invention;

FIG. 6 is an illustration showing the addition of a pass to the sequence of steps executed by the text analyzer produced by the present invention;

FIG. 7 is an illustration of a parse tree data structure modified by a partial analysis step performed by the present invention;

FIG. 8 is an illustration of an updated sample hierarchy;

FIG. 9 shows the addition of a second pass to the sequence of steps executed by the analyzer;

FIG. 10 is an illustration of a parse tree data structure modified by a second partial analysis step performed by the present invention;

FIG. 11A illustrates the relationship of various types of rules in the generator program;

FIG. 11B illustrates the logical sequence of steps for generalizing and merging rules;

FIG. 12 illustrates a user interface that allows a user to operate the generator program;

FIG. 13 illustrates the association of a text sample with the sample hierarchy;

FIG. 14 illustrates how a user labels the components a sample via the user interface;

FIG. 15 illustrates a form tool used in connection with the generator program;

FIG. 16 illustrates the properties window used in connection with the user interface;

FIG. 17 illustrates the attributes window used in connection with the user interface; and

FIG. 18 illustrates a menu incorporated into the sample manager for managing samples and integrating them with the text analyzer development environment;

FIG. 19 illustrates a sequence of steps placed into a stub region by the generator, along with the

rules generated for one of the steps.

DETAILED DESCRIPTION

Directing attention to the drawings, FIG.1 is a high level block diagram of the hardware typically used in an embodiment of the present invention. Computer 100 may have a conventional design, incorporating a processor 102 utilizing a central processing unit (CPU) and supporting integrated circuitry. Memory 104 may include RAM and NVRAM such as flash memory, to facilitate storage of computer programs executed by processor 102, such as generator program 106. Also included in computer 100 are keyboard 108, pointing device 110, and monitor 112, which allow a user to interact with program 106 during execution. Mass storage devices such as disk drive 114 and CD ROM 116 may also be incorporated in the computer 100 to provide storage for generator program 106 and associated files. Computer 100 may communicate with other computers via modem 118 and telephone line 120 to allow generator program 106 to be operated remotely, or utilize files stored at different locations. Other media may also be used in place of modem 118 and telephone line 120, such as a direct connection or high speed data line. The components described above may be operatively connected by a communications bus 122.

Generator program 106 produces text analyzer programs by generating rules from samples supplied by users to create individual passes of a multi-pass text analyzer. A sample is a piece of text that users have decided is a unit of interest, such as a name or idiomatic phrase. A sample hierarchy is an index for storing all user-added samples. A rule is a representation for a pattern of interest, which may include associated actions to ensure that the pattern has matched correctly and to record the match in the parse tree. A rule typically associates a concept with a pattern or phrase. When the pattern matches a list of nodes, the matched nodes of the parse tree are condensed or reduced to a node associated with the concept.

As used herein, a pass is one step of a multi-step analyzer, in which the generator program 106 traverses a parse tree to execute a set of rules associated with the pass. As used herein, a parse tree is a tree data structure constructed and maintained by the generator program 106 to organize text and all the patterns that have been recognized within the text. Successive passes are created in a cascading fashion by performing partial text analyses employing existing passes. The resulting text analyzer program interleaves the generated passes with a framework of existing passes. The complete text analysis system can then process text to identify patterns similar to samples added by users. Generation of rules from samples encompasses a wide range of constructs and granularities that occur in text, from individual words to intrasentential patterns (such as a grammar), to sentential, paragraph, section, and other formats that occur in text documents.

To exemplify the methods and data structures of the present invention, we use simple telephone number patterns such as

497-5318
(949) 497-5318
Home: (949) 497-5318 (1)

FIG. 2 shows a resulting text analyzer program produced by an embodiment of the present invention. Text analyzer program 200 contains three passes. The first pass, tokenize (202), processes an input text to group the characters according to alphabetic, numeric, white space, and punctuation units, referred to herein as tokens. The tokens are all placed into a parse tree data structure 300 (FIG. 3). The parse tree 300 is used and modified by subsequent passes. The phrases pass (204) is a stub, or empty placeholder, for the passes that the generator program 106 creates using user-supplied samples in a sample hierarchy. Since there are no passes there initially, this placeholder pass has no effect on the parse tree 300. Finally, the output pass (206), displays a representation of the parse tree 300.

Given the sample input text:

The output pass displays the parse tree 300 as illustrated in FIG. 3.

Text analyzer program 200 has no knowledge of telephone number patterns. If a user wants phone numbers to be grouped under a concept called phone, a sample hierarchy as shown in FIG. 4 can be constructed. This hierarchy of samples 400 has a top-level stub concept called phrases 402, which matches the stub region 204 to be filled within the text analyzer 200. The user creates a rule concept called phone 404, in order to place telephone number samples under it. Under the rule concept 404, the user places telephone number samples such as "497-5318" and labels their components with the arbitrary names prefix 406 corresponding to "497" and suffix 408 corresponding to "5318," which are referred to herein as label concepts 410. Such samples can be added in a simple fashion with a user interface that allows highlighting the complete text and its components.

Generator program 106 can be invoked to generate a new analyzer by executing the sequence of steps 500 illustrated in the FIG. 5. Generator program 106 first traverses the sample hierarchy 400 to find the rule concept called phone 404 (step 502). If a rule concept is found (decision step 504), it traverses the samples corresponding to the rule concept found, encountering the phone number 497-5318 (step 506). The generator program 106 executes a partial analysis (step 508) of the text containing the current sample. Because a pass will be generated for the current rule concept, the partial analysis stops just before the pass to be created. (See decision step 510.) In our simplified example, the partial analysis consists only of the tokenize pass 202, so that is what the generator program 106 executes. Partial analysis involves applying the partially built text analyzer, containing passes constructed so far, to the text containing the samples used to build the current pass. Partial analysis is conducted in an iterative fashion. Continuing to step 514, the generator program 106 locates the position of the current sample within the parse tree that has been constructed so far (FIG. 3), based on the offsets of the sample within its entire text (step 512). Going back to the

example, "(949) 497-5318" has a start and end offset in the text that it appears in, for example 0 to 13, if it is in the beginning of a text file. Looking in the parse tree later in the parse, the phrase that covers this range of offsets now appears "(949) _phone." Each part or node of the parse tree has a start and end offset. The "_phone" portion accounts for characters 6 through 13. The generator program 106, in building a complete phone rule, uses whatever phrase it finds in the parse tree 300 at the range of offsets from 0 to 13. In the parse tree 300, the generator program 106 finds the tokens

497 \- 5318 (3)

It therefore generates [step 516] the raw rule based precisely on what is represented in the parse tree, as follows:

_phone <- 497 \- 5318 @@ (4)

The underscore before phone indicates that this is a non-literal concept. The <- arrow indicates a rewrite of the phrase to the right with the concept to the left. The @@ marker denotes the end of the rule. The backslash preceding the dash means that this dash is to be taken literally, rather than being part of the rule language. At this point, the generator program 106 can attach labeling information to the first element ("497") and the last element ("5318") of the phrase, as prefix and suffix, as follows:

_phone <- 497 [label=_prefix] \- 5318 [label=_suffix] @@ (5)

Since there are no other samples (decision step 518) under the phone concept, the generator program 106 has no opportunity to merge and compare samples. Having finished with the samples under this rule concept, the generator program 106 at step 526 creates a new pass called phone for the rule set it has generated (consisting of one rule in this case). The generator program 106 then

adds the new pass to the analyzer sequence (step 528), as shown in FIG. 6. The generator program 106 then looks for the next rule concept at step 530. Had there been additional rule concepts with samples in the sample hierarchy, (step 520) control would return to step 508, where the generator program 106 would have proceeded to analyze the overall text from which those samples derived. It would perform a partial analysis up to and including the pass called phone. For the given text, the resulting parse tree data structure 700 is shown in FIG. 7. Note that the tokens "497", "-", and "5318" have been replaced in the parse tree by the single token "_phone". Now let us suppose that the user adds a second sample to the hierarchy, under a new concept, yielding the sample hierarchy shown in FIG. 8.

Had the phone concept not been in this sample hierarchy, the generator program 106 would have built the rule

_completePhone <- \(949 [label=areaCode] \) \ 497 \- 5318 @@ (6)

But because the phone sample is also present and the generator program 106 has installed the phone pass within the analyzer, the generator program 106 is given parse tree 550 (FIG. 7) when constructing the rule for completePhone. Therefore, the generator program 106 builds the following rule:

_completePhone <- \(949 [label=areaCode] \) \ _phone @@ (7)

The product of the prior automatically-generated pass is used in building the rules for the current pass called completePhone. The generator program 106 has now built an analyzer for phone numbers that follows the passes illustrated in FIG. 9. In this example, the phone pass and completePhone pass each contain one rule. This analyzer with two automatically generated passes produces the parse tree in FIG. 10 from the sample input text in (2).

Generator program 106 automatically creates the passes of a text analyzer in stepwise

fashion, each time using the sequence of passes constructed so far in order to create the next pass of the analyzer. It adds each new pass to a backbone of manually built and previously generated passes.

The discussion above describes the generation of one pass per rule concept. Additional modes, specified by the user who constructs the sample hierarchy, enable the rules generated for multiple rule concepts to be merged into a single large pass (step 524), in order to both optimize performance and to enable more sophisticated rule generation that identifies and unifies ambiguous constructs. For example, if "New York" is listed under a rule concept city and a rule concept state, then a unified treatment of these rule concepts can enable the generation of rules such as:

_city [label=_state] <- New York @@ (8)

which condenses instances of "New York" to both a city concept and a state concept in a parse tree.

Optimizations

Executing the generator program 106 can be computationally expensive, because each sample in the sample hierarchy requires the text containing it to be partially analyzed, in order to generate the rule corresponding to the sample. Generator program 106 can be modified to keep track of instances where multiple samples under a rule concept derive from the same text. In those cases, the given text need be partially analyzed only once, in order to glean the RAW rules for all the samples that derived from that text.

In a preferred embodiment, further optimization may be achieved when generator program 106 places user-added samples into a single sample file. Thus, each rule file has an associated sample file. The sample file may be stored in memory 104, disk drive 114 or CD Rom 116. In this way the number of partial text analyses is reduced for a sample hierarchy with many samples. Further optimizations are to generate passes when their complement of samples has changed. While there is a danger that some subsequent pass may not be updated correctly due to dependencies on

the current pass, most of the time this method of generation

(generate dirty) is adequate for rapid development and testing. Occasionally, a generate all function may be invoked to rebuild every single pass, thus making sure that all passes that need updating will get updated.

Rule Generalization and Merging

A preferred embodiment of the present invention also has the capability to generalize and merge raw rules generated directly from samples as illustrated in FIG. 5 at step 524. One sample is usually not sufficient to derive or generalize rules. At least two samples of any given pattern are required in order to deduce the more general pattern. When multiple samples are available under a rule concept, the rule generalization and merging method is invoked at step 524 to build a variety of rule sets: literal, general, optional, split, and constrained. The hierarchy shown in FIG. 11A, and the flow diagram in FIG. 11B, best describe the relationships among these rule sets.

At step 560, for each raw rule generated (one per sample), the generator program 106 creates a general rule by iteratively generalizing each element of the raw rule. For example, "497" will be generalized to NUMBER, "Home" will be generalized to ALPHABETIC, "-" to PUNCTUATION, and " " to WHITESPACE. At step 562, generator program 106 merges general rules that have identical elements and length. The general rule for "497-5318" will be identical to that for "555-1212," namely

_phone <- _NUMBER _PUNCTUATION _NUMBER @@ (14)

Therefore the rules for the two samples are merged under this general rule. The general rule retains a list of all the raw rules that gave rise to it. At step 564, generator program 106 traverses the general rules to build the split rules. The split rules require that all raw rules have consistent labeling. So a split rule may appear:

_phone <- _NUMBER [label=_prefix] _PUNCTUATION _NUMBER [label=_suffix] @@ (15)

At step 566, generator program traverses the split rules to generate the constrained rules. Constrained rules are rules whose raw rules all have consistent features, such as length and capitalization.

A constrained rule may appear:

```
_phone <-
        _NUMBER [label=_prefix length=3]
        _PUNCTUATION
        _NUMBER [label=_suffix length=4]
        @@                                              (16)
```

The above rule constrains the first number to have three digits and the second number to have four digits. At step 566, generator program 106 creates a literal rule for every raw rule. The literal rule is constructed by looking "inside" each element of the phrase as deeply as can be seen in the parse tree. For example, if a raw rule appears

_phone <- _LIST (NUMBER 497) \- _LIST (NUMBER 5318) @@ (17)

the literal rule produced is

_phone <- 497 \- 5318 @@ (18)

At step 570, generator program 106 creates optional rules by comparing the composition of general rules that differ by one element. If that element is not a labeled element, then the two general rules can be merged, with the difference element marked as optional.

By embellishing a sample hierarchy with particular attributes, the manner in which rules are generated is controlled. The need to collect large sample sets in order to calculate statistically plausible generalizations is eliminated. Attributes may be specified to indicate what is to be generalized, what is to be collected as a list, and what is to be retained literally. For example, one attribute may instruct the generator program 106 to always generalize whitespace to a rule element that allows an arbitrary number of space characters. Another attribute may designate a label concept

as "closed," meaning any samples within it are to be collected into a list of only those samples, with no generalization. Other flags control the rule sets to be retained for the pass being generated. If the "constrain" flag is set to "true," then the constrained list of rules is retained by the generator program 106. Retaining a rule set involves placing it into the final list of rules for the pass under construction. An enhancement to the sample hierarchy is to enable the described attributes to control the way rules are generated for an entire subtree. If some concept within that subtree changes an attribute's value, then that new value controls its subtree, and so on recursively.

A nonexhaustive set of attributes may be utilized to allow a user to control the rule sets to be retained in each pass of the analyzer, as below:

Attribute	Values
GENERAL	true/false
SPLIT	true/false
CONSTRAINED	true/false
RAW	true/false
LITERAL	true/false

(19)

The above attributes cause the generator program 106 to retain or discard the corresponding rule sets. For example, if a concept in the sample hierarchy has the constrained attribute set to true, then all the constrained rules generated in that subhierarchy will be retained as part of the final analyzer. An attribute called closed, also with true/false values, controls the way parts of samples are collected into rules. For example, given the samples

497-5318
555-1212 (20)

if the closed attribute is set to true, then the corresponding constrained phone rule appears

_phone <- _LIST (497 555) \- _LIST (5318 1212) @@ (21)

That is, each element of the pattern is a "closed set," which collects any values found in the set of samples. If the CLOSED attribute is set to false, the constrained rule is

```
_phone <-
        _NUMBER [label=_prefix length=3]
        _PUNCTUATION
        _NUMBER [label=_suffix length=4]
        @@                                                    (22)
```

Because white space and punctuation are often secondary in importance, a WHITE attribute with values true/false can specify that whitespace in samples generalizes to the rule element

```
_WHITE [min=0 max=infinity]                                   (23)
```

that is, any number of white spaces, regardless of the particular type and number of whitespace characters in the set of samples.

Other attributes can control the actions that get built for the generated rules and their components. For example, a QUICKSEM attribute with values true/false generates actions for semantic information to be copied automatically when a rule matches text. In the phone number example, the QUICKSEM attribute would cause the automatic creation of a data item called "prefix" with value "497" and a second data item called "suffix" with value "5318" in the _phone node, given that the _phone rule matched a text string such as "497-5318." The LABEL (or LAYER) attribute takes a name as its value and leads to the generation of a label action in the associated rules that get generated.

USER INTERFACE

FIG. 12 illustrates a user interface 600 that allows a user to operate the generator program 106. The left panel 602 displays the sample hierarchy 604 with phoneNumber concept 606 selected. The right panel 608 displays the rule file automatically generated for the phoneNumber concept. The partial hierarchy file listing below details the commands for building the concepts of the sample hierarchy for a generator program 106 configured to process resumes. Each line builds one concept (the listing does not distinguish among organizing concepts, rule concepts, and label concepts). Each concept can have an arbitrary number of samples assigned to it by a user. While most of the samples for the generator program 106 are smaller than a sentence (intrasentential), the method and

system of the present invention apply to paragraphs and sections of texts as well as to intrasentential patterns.

"concept" "gram" "LiteralPhrase" "HeaderPhrase"

"concept" "gram" "LiteralPhrase" "HeaderPhrase" "ContactHeaderPhrase"

"concept" "gram" "LiteralPhrase" "HeaderPhrase" "ObjectiveHeaderPhrase"

"concept" "gram" "LiteralPhrase" "HeaderPhrase" "EducationHeaderPhrase"

"concept" "gram" "LiteralPhrase" "HeaderPhrase" "ExperienceHeaderPhrase"

"concept" "gram" "LiteralPhrase" "HeaderPhrase" "SkillsHeaderPhrase"

"concept" "gram" "LiteralPhrase" "HeaderPhrase" "PresentationsHeaderPhrase"

"concept" "gram" "LiteralPhrase" "HeaderPhrase" "PublicationsHeaderPhrase"

"concept" "gram" "LiteralPhrase" "HeaderPhrase" "ReferencesHeaderPhrase"

"concept" "gram" "LiteralPhrase" "HeaderPhrase" "OtherHeaderPhrase"

"concept" "gram" "LiteralPhrase" "Others"

"concept" "gram" "LiteralPhrase" "Others" "degreeInMajor"

"concept" "gram" "LiteralPhrase" "Others" "WebLinks"

"concept" "gram" "LiteralPhrase" "Others" "emailHeader"

"concept" "gram" "LiteralPhrase" "Others" "minorKey"

"concept" "gram" "LiteralPhrase" "Caps"

"concept" "gram" "LiteralPhrase" "Caps" "cityPhrase"

"concept" "gram" "LiteralPhrase" "Caps" "statePhrase"

"concept" "gram" "LiteralPhrase" "Caps" "companyPhrase"

"concept" "gram" "LiteralPhrase" "Caps" "degreePhrase"

"concept" "gram" "LiteralPhrase" "Caps" "countryPhrase"

"concept" "gram" "LiteralPhrase" "Caps" "skillsPhrase"

"concept" "gram" "LiteralPhrase" "Caps" "naturalLanguages"

"concept" "gram" "LiteralPhrase" "Caps" "software"

"concept" "gram" "LiteralPhrase" "Caps" "hardware"

"concept" "gram" "LiteralPhrase" "Caps" "certifications"

"concept" "gram" "LiteralPhrase" "Caps" "field"

"concept" "gram" "LiteralPhrase" "Caps" "Thesis"

"concept" "gram" "LiteralPhrase" "Caps" "jobTitle"

"concept" "gram" "LiteralPhrase" "Caps" "jobPhrase"

"concept" "gram" "Word"

"concept" "gram" "Word" "Syntax"

"concept" "gram" "Word" "Syntax" "posPREP"

"concept" "gram" "Word" "Syntax" "posDET"

"concept" "gram" "Word" "Syntax" "posPRO"

"concept" "gram" "Word" "Syntax" "posCONJ"

"concept" "gram" "Word" "HeaderWord"

"concept" "gram" "Word" "HeaderWord" "ContactHeaderWord"

"concept" "gram" "Word" "HeaderWord" "ObjectiveHeaderWord"

"concept" "gram" "Word" "HeaderWord" "EducationHeaderWord"

"concept" "gram" "Word" "HeaderWord" "ExperienceHeaderWord"

"concept" "gram" "Word" "HeaderWord" "SkillsHeaderWord"

"concept" "gram" "Word" "HeaderWord" "PresentationsHeaderWord"

"concept" "gram" "Word" "HeaderWord" "PublicationsHeaderWord"

"concept" "gram" "Word" "HeaderWord" "ReferencesHeaderWord"

"concept" "gram" "Word" "HeaderWord" "OtherHeaderWord"

"concept" "gram" "Word" "headerMod"

"concept" "gram" "Word" "openPunct"

"concept" "gram" "Word" "closePunct"

"concept" "gram" "Word" "resumeWord"

"concept" "gram" "Word" "Present"

"concept" "gram" "Word" "Direction"

"concept" "gram" "Word" "adjDirection"

"concept" "gram" "Word" "PostalUnit"

"concept" "gram" "Word" "PostalRoad"

"concept" "gram" "Word" "monthWord"

"concept" "gram" "Word" "monthNum"

"concept" "gram" "Word" "Season"

"concept" "gram" "Word" "PostalState"

"concept" "gram" "Word" "jobTitleRoot"

"concept" "gram" "Word" "jobMod"

"concept" "gram" "Word" "companyRoot"

"concept" "gram" "Word" "companyModroot"

"concept" "gram" "Word" "companyMod"

"concept" "gram" "Word" "ProgrammingLanguage"

"concept" "gram" "Word" "cityMod"

"concept" "gram" "Word" "cityWord"

"concept" "gram" "Word" "Names"

"concept" "gram" "Word" "Names" "femaleName"

"concept" "gram" "Word" "Names" "maleName"

"concept" "gram" "Word" "Names" "surName"

"concept" "gram" "Word" "fieldName"

"concept" "gram" "Word" "subOrg"

"concept" "gram" "Word" "softwareWord"

"concept" "gram" "Phrase"

"concept" "gram" "Phrase" "Contact"

"concept" "gram" "Phrase" "Contact" "humanName"

"concept" "gram" "Phrase" "Contact" "humanName" "prefixName"

"concept" "gram" "Phrase" "Contact" "humanName" "firstName"

"concept" "gram" "Phrase" "Contact" "humanName" "middleName"

"concept" "gram" "Phrase" "Contact" "humanName" "lastName"

"concept" "gram" "Phrase" "Contact" "humanName" "suffixName"

"concept" "gram" "Phrase" "Contact" "cityStateZip"

"concept" "gram" "Phrase" "Contact" "cityStateZip" "cityName"

"concept" "gram" "Phrase" "Contact" "cityStateZip" "stateName"

"concept" "gram" "Phrase" "Contact" "cityStateZip" "zipCode"

"concept" "gram" "Phrase" "Contact" "cityStateZip" "zipSuffix"

"concept" "gram" "Phrase" "Contact" "cityStateZip" "country"

"concept" "gram" "Phrase" "Contact" "cityState"

"concept" "gram" "Phrase" "Contact" "cityState" "cityName"

"concept" "gram" "Phrase" "Contact" "cityState" "stateName"

"concept" "gram" "Phrase" "Contact" "phoneExtension"

"concept" "gram" "Phrase" "Contact" "phoneExtension" "extendWord"

"concept" "gram" "Phrase" "Contact" "phoneExtension" "extension"

"concept" "gram" "Phrase" "Contact" "phoneNumber"

"concept" "gram" "Phrase" "Contact" "phoneNumber" "countryCode"

"concept" "gram" "Phrase" "Contact" "phoneNumber" "areaCode"

"concept" "gram" "Phrase" "Contact" "phoneNumber" "prefix"

"concept" "gram" "Phrase" "Contact" "phoneNumber" "suffix"

"concept" "gram" "Phrase" "Contact" "phonePhrases"

"concept" "gram" "Phrase" "Contact" "phonePhrases" "phoneHomeFaxPhrase"

"concept" "gram" "Phrase" "Contact" "phonePhrases" "phoneHomeFaxPhrase" "HomeFax"

"concept" "gram" "Phrase" "Contact" "phonePhrases" "phoneWorkPhrase"

"concept" "gram" "Phrase" "Contact" "phonePhrases" "phoneWorkPhrase" "Work"

"concept" "gram" "Phrase" "Contact" "phonePhrases" "phoneFaxPhrase"

"concept" "gram" "Phrase" "Contact" "phonePhrases" "phoneFaxPhrase" "Fax"

"concept" "gram" "Phrase" "Contact" "phonePhrases" "phonePagerPhrase"

"concept" "gram" "Phrase" "Contact" "phonePhrases" "phonePagerPhrase" "Pager"

"concept" "gram" "Phrase" "Contact" "phonePhrases" "phoneCellPhrase"

"concept" "gram" "Phrase" "Contact" "phonePhrases" "phoneCellPhrase" "Cell"

"concept" "gram" "Phrase" "Contact" "phonePhrases" "phoneHomePhrase"

"concept" "gram" "Phrase" "Contact" "phonePhrases" "phoneHomePhrase" "Home"

"concept" "gram" "Phrase" "Contact" "unitRoom"

"concept" "gram" "Phrase" "Contact" "unitRoom" "unit"

"concept" "gram" "Phrase" "Contact" "unitRoom" "room"

"concept" "gram" "Phrase" "Contact" "addressLine"

"concept" "gram" "Phrase" "Contact" "addressLine" "streetNumber"

"concept" "gram" "Phrase" "Contact" "addressLine" "streetName"

"concept" "gram" "Phrase" "Contact" "addressLine" "road"

"concept" "gram" "Phrase" "Contact" "addressLine" "direction"

"concept" "gram" "Phrase" "Contact" "addressLine" "postdirection"

"concept" "gram" "Phrase" "Contact" "addressLine" "POBox"

"concept" "gram" "Phrase" "Contact" "email"

"concept" "gram" "Phrase" "Contact" "email" "accountName"

"concept" "gram" "Phrase" "Contact" "email" "machineName"

"concept" "gram" "Phrase" "Contact" "email" "companyName"

"concept" "gram" "Phrase" "Contact" "email" "domainName"

"concept" "gram" "Phrase" "Contact" "url"

"concept" "gram" "Phrase" "Contact" "url" "urlHeader"

"concept" "gram" "Phrase" "Contact" "url" "protocol"

"concept" "gram" "Phrase" "Contact" "url" "machineName"

"concept" "gram" "Phrase" "Contact" "url" "companyName"

"concept" "gram" "Phrase" "Contact" "url" "domainName"

"concept" "gram" "Phrase" "Contact" "url" "directory"

"concept" "gram" "Phrase" "Contact" "Height"

"concept" "gram" "Phrase" "Contact" "Height" "feet"

"concept" "gram" "Phrase" "Contact" "Height" "inches"

"concept" "gram" "Phrase" "Education"

"concept" "gram" "Phrase" "Education" "degree"

"concept" "gram" "Phrase" "Education" "major"

"concept" "gram" "Phrase" "Education" "minor"

"concept" "gram" "Phrase" "Education" "university"

"concept" "gram" "Phrase" "Experience"

"concept" "gram" "Phrase" "Experience" "company"

"concept" "gram" "Phrase" "SingleDate"

"concept" "gram" "Phrase" "SingleDate" "numA"

"concept" "gram" "Phrase" "SingleDate" "numB"

"concept" "gram" "Phrase" "SingleDate" "monthSD"

"concept" "gram" "Phrase" "SingleDate" "daySD"

"concept" "gram" "Phrase" "SingleDate" "yearSD"

"concept" "gram" "Phrase" "SingleDate" "seasonSD"

"concept" "gram" "Phrase" "DateRange"

"concept" "gram" "Phrase" "DateRange" "fromDate"

"concept" "gram" "Phrase" "DateRange" "dateSep"

"concept" "gram" "Phrase" "DateRange" "toDate"

"concept" "gram" "Part"

"concept" "gram" "Part" "addressPart"

"concept" "gram" "Part" "educationPart"

"concept" "gram" "Part" "experiencePart"

Machinery for Adding and Managing Samples

While a command line interface may be utilized by an embodiment of the present invention, the preferred embodiment utilizes a graphical user interface (GUI) to manage the sample hierarchy. A specialized pull-down menu enables rapid highlighting and labeling of samples and their components within a text. By selecting a concept in the sample hierarchy and then highlighting a text, the highlighted text sample is placed under the sample hierarchy concept, as in FIG. 13. Once the user adds the overall sample, he can proceed to add labels (i.e., components of the overall sample), as illustrated in FIG. 14. As shown in FIG. 14, the user highlights and labels "Long Beach" as a cityName.

In another aspect of the user interface of the present invention, a form tool 580 (FIG. 15) accelerates and organizes the addition of a sample and labeling of its components by enabling a user to quickly group the textual components of a sample so that they will be properly labeled. Form tool

580 minimizes the need to use the mouse for highlighting a sample text and each of its components. Merely by clicking arrows in the form tool 580, the user can rearrange the components of the overall sample so that they are grouped and labeled properly. The form tool 580 can also serve as a locus for specifying information about a sample and for guiding the generation of rules and actions from the sample.

Additional tools associated with the sample hierarchy are the Attribute Window and the Properties Window. The Properties Window 582 (FIG. 16) provides a structured way to control the mode of generating rules for a subhierarchy of the sample hierarchy. The Attribute Window 584 (FIG. 17) is a lower level interface to attributes, enabling the user to add attributes that have not yet been incorporated into the Properties Window 582.

Sample manager 586 is responsible for bookkeeping to track the file that each sample originated from and the offsets of the sample and its labels within that file. The user may further associate a sample file with any concept in the sample hierarchy. If the user creates such an association, then the system creates copies of samples, their labels, and their offsets in the sample file. Sample files enable faster and more efficient generation of the text analyzer by minimizing the volume of text that must be analyzed to generate the rules for the analyzer. The sample manager 586 enables the user to perform functions such as associating a sample file, dissociating a sample file, opening the associated sample file, deleting the samples under a concept, and similar manipulations. FIG. 18 illustrates a menu incorporated into the sample manager 586 for managing samples and integrating them with the text analyzer development environment. The major capabilities available to the user in the sample manager 586 include:

FUNCTION	DESCRIPTION
Add Concept	Add a concept to the sample hierarchy, under selected concept.
Add Top Concept	Add a top-level concept to the sample hierarchy.
Add Stub	Add a top-level concept and link it to a region of the text analyzer

sequence.

Delete	Delete the selected item in the sample hierarchy.
Delete Children	Delete the children of the selected concept.
Find	Find the selected concept or sample by name.
Associate Sample File	Associate a sample file with the selected concept.
Convert to Sample File	Write the samples under selected concept to a sample file.
Delete Samples	Delete the samples under selected concept.
Disassociate File	Disassociate the sample file from the selected concept.
Generate Sample File	Generate sample file for selected concept.
Open Sample File	Open a sample file for study or editing
Attributes	Bring up the Attribute Window.
Highlight Matches	Show where a concept's rules have matched an analyzed text.
Mark for Generation	Mark concept for quick generation of rules (i.e., generate dirty).
Properties	Bring up the Properties Window.
Rules	Edit or view the Rule File generated for the selected concept.
View Tree	View the parse tree due to selected concept's rules (for analyzed text).
Archive Grammar	Store the sample hierarchy in the local archive.
Local Archive	View the local archive.
Server Archive	View the remote archive.
Upload Grammar	Store the sample hierarchy in the remote archive.

The left panel 590 in FIG. 19 illustrates the automatically generated passes in the analyzer sequence, corresponding to the Phrase Stub 588 of FIG. 18. The right panel 592 shows the selected file, phoneNumber, within the stub region.

We have described a system, method, and computer readable medium for generating text analyzers from samples. The users of a text analyzer need not understand how rules are generated

in order to maintain and enhance the capabilities of the text analyzer. Nonprogrammer and nonlinguist users can add samples that the text analyzer does not identify, in order to expand the processing power of the text analyzer. While the present invention has been illustrated and described in detail, it is to be understood that numerous modifications may be made to the preferred embodiment without departing from the spirit of the invention.

CLAIMS

What is claimed is:

1. A method for generating a text analysis program for recognizing patterns appearing in text and extracting information from said patterns, the method comprising the steps of:

(a) providing a sample hierarchy, said sample hierarchy comprising samples of text;

(b) extracting at least one rule from said sample hierarchy, said rule describing how to process a portion of text;

(c) generating a pass from said rule, said pass containing instructions to operate a text analyzer; and

(d) constructing a text analyzer containing said pass.

2. The method of claim 1, wherein said rule is generalized into multiple rules and multiple passes.

3. The method of Claim 1, wherein multiple passes are added to said text analyzer.

4. The method of Claim 3, wherein said multiple passes are arranged in a cascading manner having a sequence of passes such that rules associated with a pass are applied to subsequent passes.

5. The method of Claim 1, wherein the samples are associated with offset values, said offset values identifying locations in a parse tree data structure, said parse tree containing concepts stored at locations identified by said offsets.

6. The method of Claim 4, further comprising the step of allowing a user to control the extraction of rules from the sample hierarchy

7. The method of Claim 5, further comprising the step of allowing a user to designate properties

associated with said rules, said properties controlling rule generation for a portion of the sample hierarchy.

8. The method of Claim 5, wherein said concepts are retrieved from said parse tree and processed to form said rule.

9. The method of Claim 6, further comprising the step of allowing a user to designate attributes associated with said rules, said attributes guiding the application of said rules.

10. The method of Claim 1, wherein multiple rules are generalized and merged into a single rule if there is a difference between the multiple rules.

11. The method of Claim 10, wherein said samples may be contained in a sample file.

12. A sample hierarchy data structure for use in a text analyzer system, said sample hierarchy comprising an index for storing samples, said samples comprising portions of text, said samples used to generate rules for identifying patterns appearing in text, said samples used to derive information from said identified patterns, said rules generated by parsing said text samples, said index organized such that passes comprising operational steps and rules are generated in an order wherein simple patterns are recognized by said text analyzer, and said recognized simple patterns are used by said text analyzer system and used to iteratively recognize more complex patterns.

13. A computer readable medium containing instructions which, when executed by a computer, generate a text analysis program for recognizing patterns appearing in text and extracting information from said patterns, by:

(a) providing a sample hierarchy, said sample hierarchy comprising samples of text;

(b) extracting at least one rule from said sample hierarchy, said rule describing how to process a portion of text;

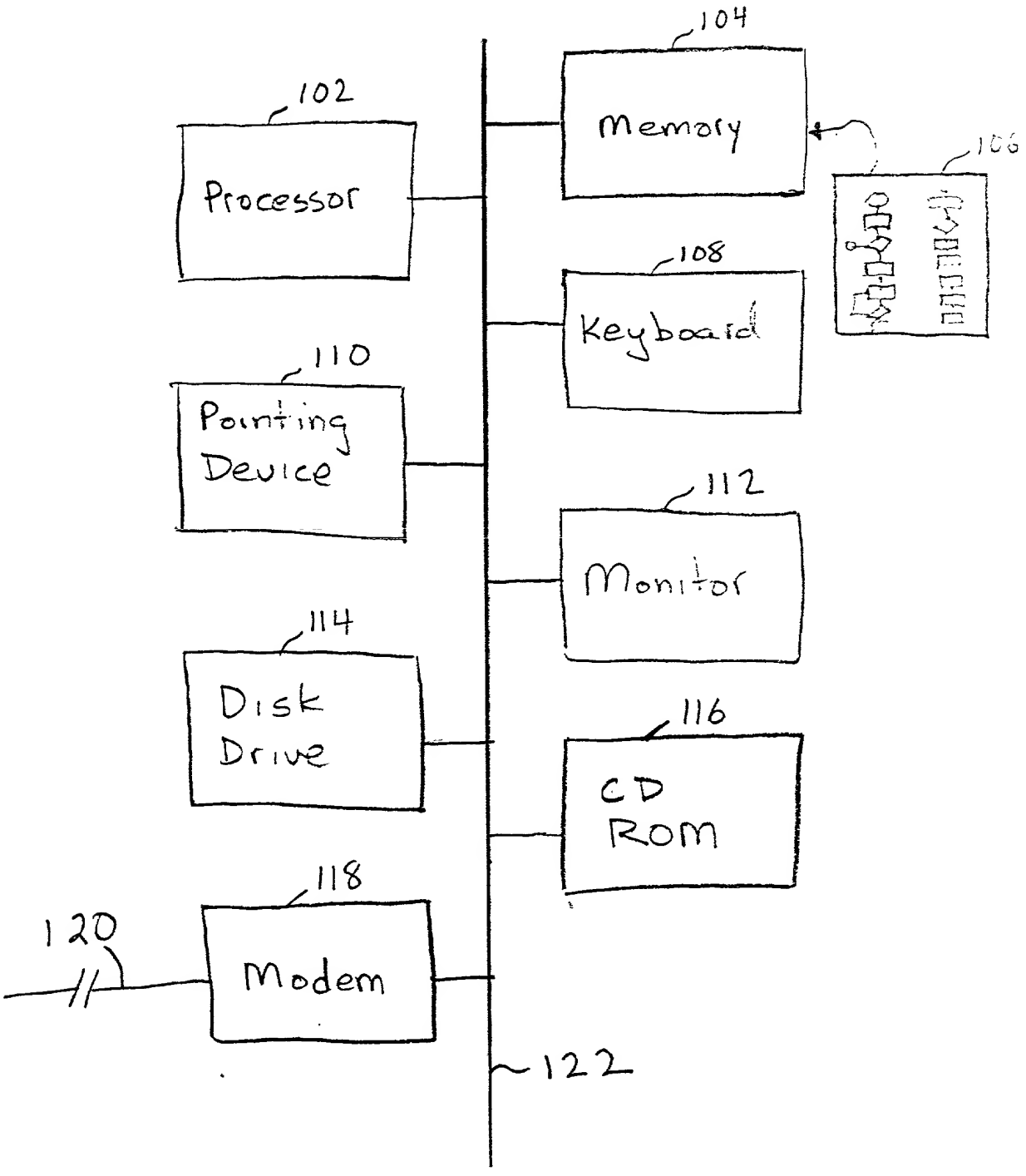
(c) generating a pass from said rule, said pass containing instructions to operate a text analyzer; and

(d) constructing a text analyzer containing said pass.

ABSTRACT

A system, method, and computer program for automatically generating text analysis systems is disclosed. Individual passes of a multi-pass text analyzer are created by generating rules from samples supplied by users. Successive passes are created in a cascading fashion by performing partial text analyses employing existing passes. A complete text analyzer interleaves the generated passes with a framework of existing passes. The complete text analysis system can then process texts to identify patterns similar to samples added by users. Generation of rules from samples encompasses a wide range of constructs and granularities that occur in text, from individual words to intrasentential patterns, to sentential, paragraph, section, and other formats that occur in text documents.

FIG. 1


104
Memory
102
Processor
106
108
Keyboard
110
Pointing Device
112
Monitor
114
Disk Drive
116
CD ROM
118
120
Modem
122

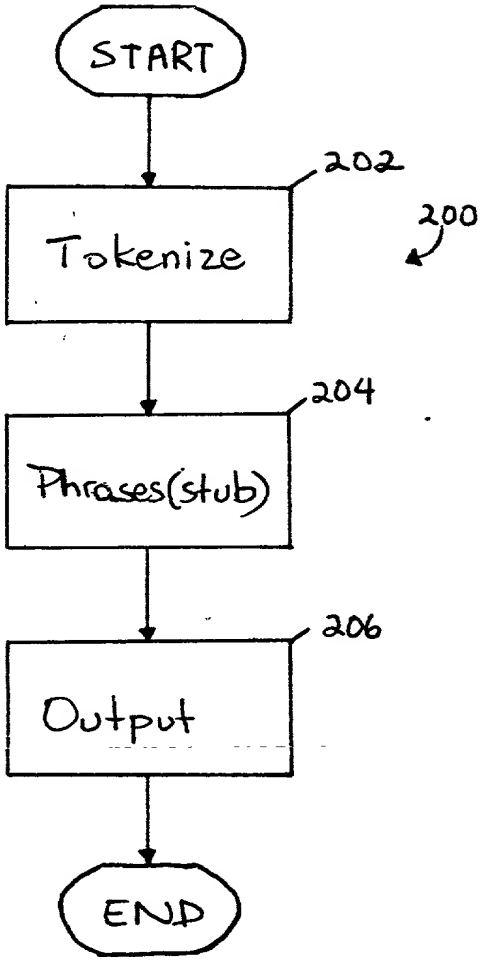


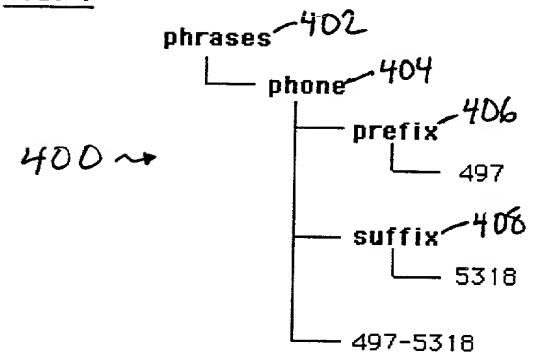
FIG. 2

```
ROOT
        Home
        \:
        \   # space
        \(
        949
        \)
        \   # space
        497
        \-
        5318
```

300

FIG. 3

FIG. 4




```
tokenize
phrases (stub)
        phone
output
```

FIG. 6

```
ROOT
        Home
        \:
        \     # space
        \(
        949
        \)
        \     # space
        _phone
          _prefix
                497
          \-
          _suffix
                5318
```

FIG. 7

FIG. 5

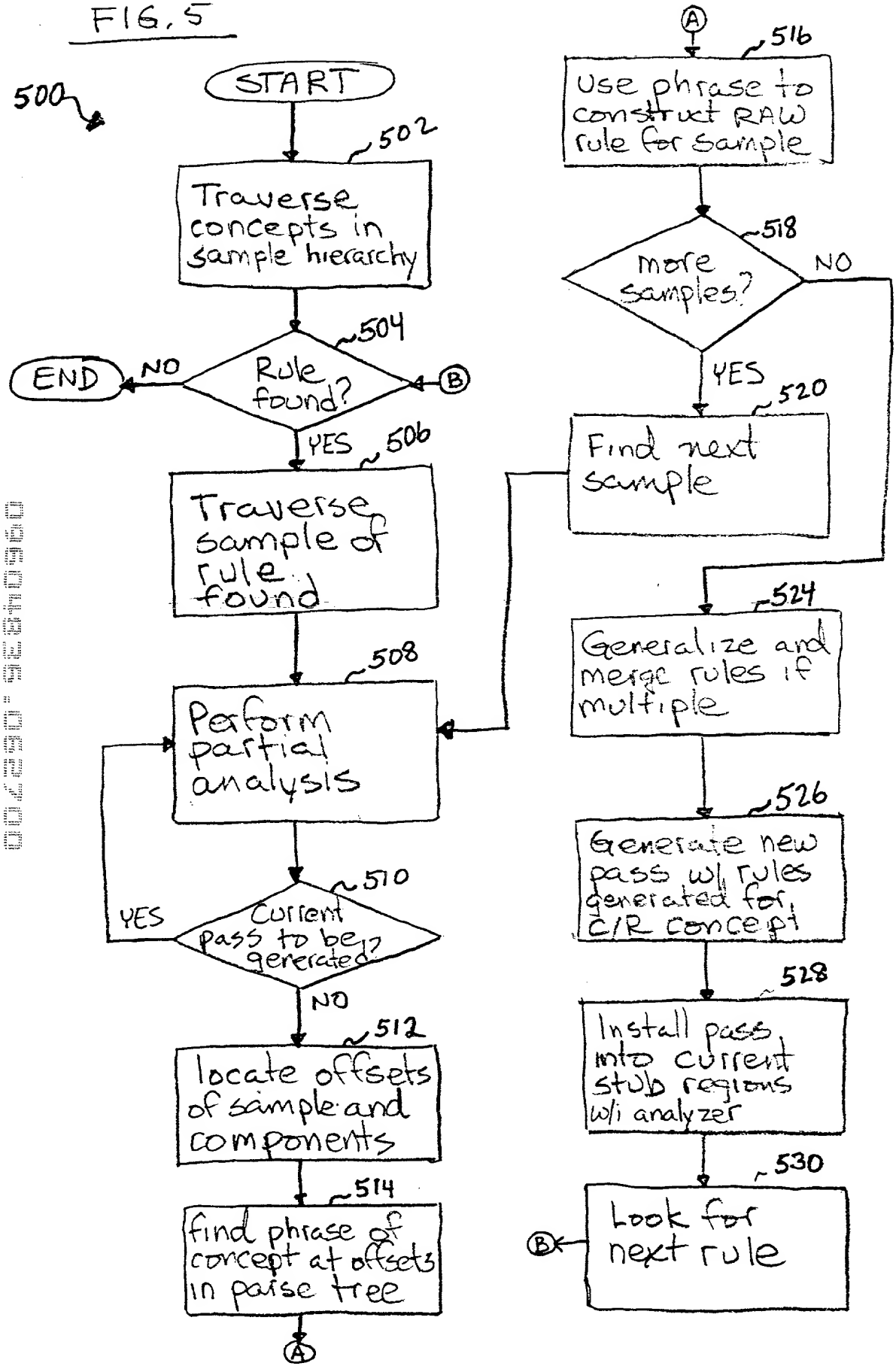
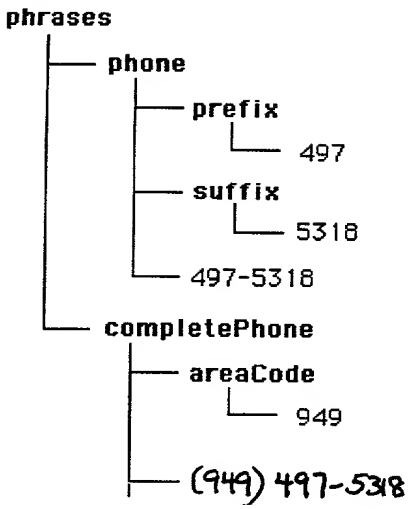

500
START
502
Traverse concepts in sample hierarchy
504
Rule found?
NO
END
B
YES
506
Traverse sample of rule found
508
Perform partial analysis
510
Current pass to be gererated?
YES
NO
512
locate offsets of sample and components
514
find phrase of concept at offsets in parse tree
A
A
516
Use phrase to construct RAW rule for sample
518
more samples?
NO
YES
520
Find next sample
524
Generalize and merge rules if multiple
526
Generate new pass w/ rules generated for C/R concept
528
Install pass into current stub regions w/ analyzer
530
Look for next rule
B

FIG. 8




```
tokenize
phrases (stub)
        phone
        completePhone
output
```

FIG. 9

```
ROOT
        Home
        \:
        \     # space
        _completePhone
                \(
                _areaCode
                        949
                \)
                \     # space
                _phone
                        _prefix
                                497
                        \-
                        _suffix
                                5318
```

FIG. 10

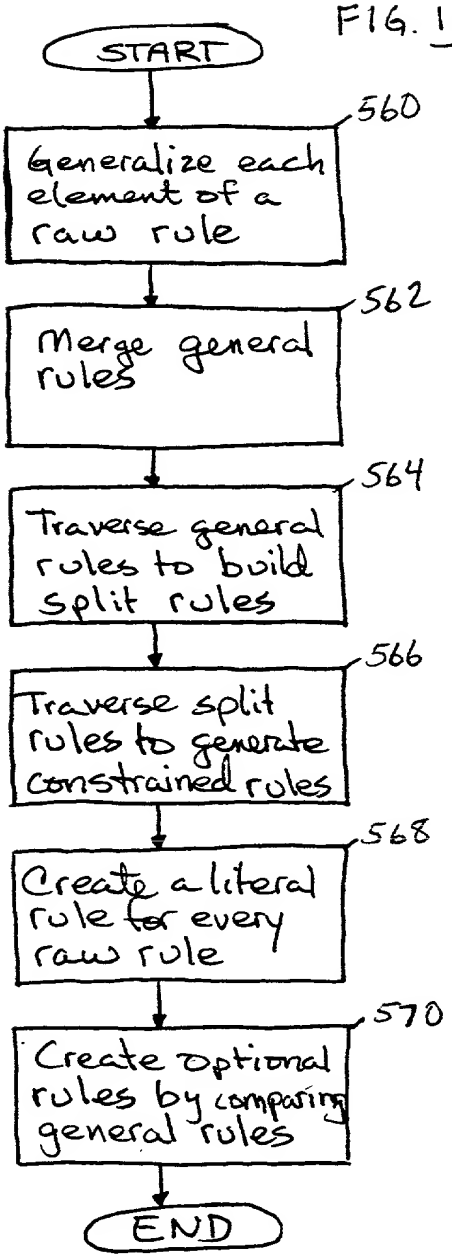


FIG. 11B

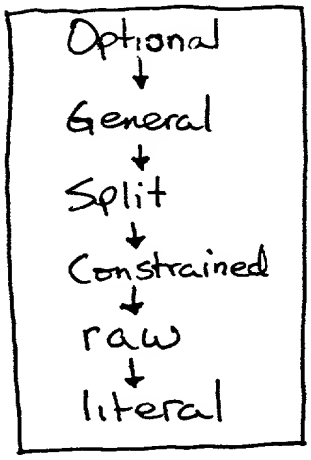


FIG. 11A

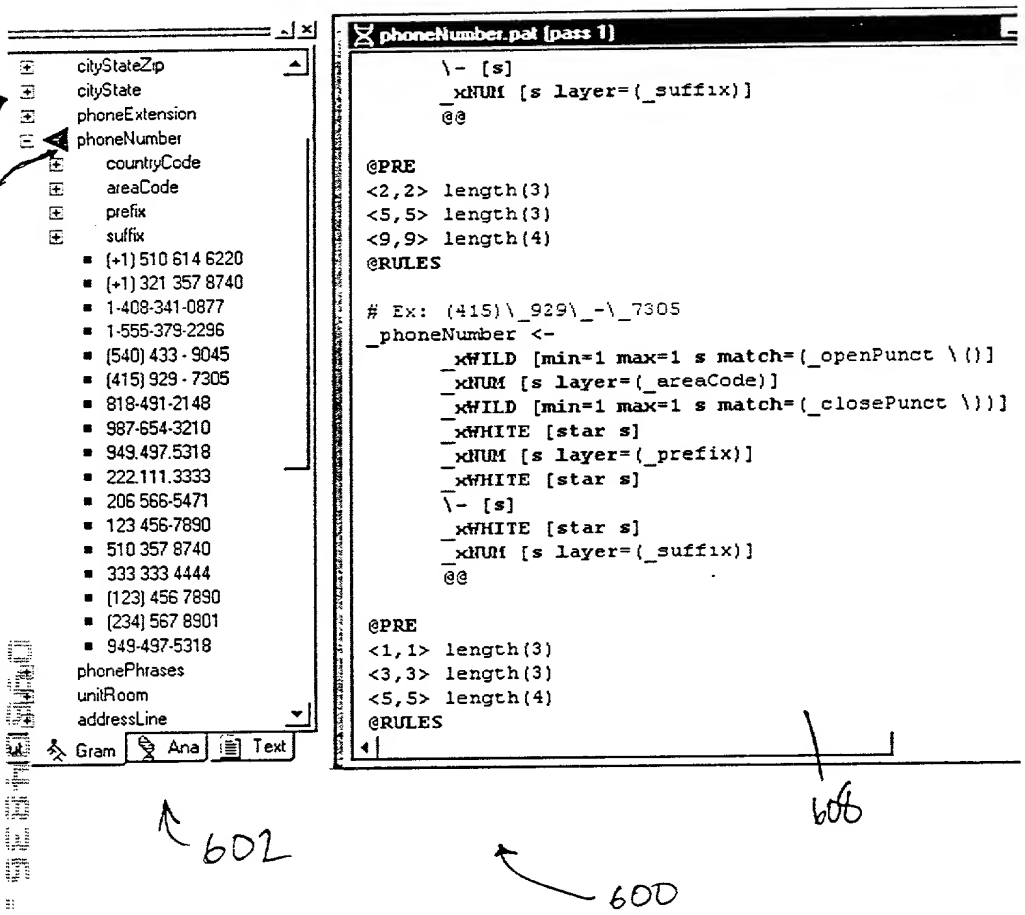

cityStateZip
cityState
phoneExtension
phoneNumber
countryCode
areaCode
prefix
suffix
(+1) 510 614 6220
(+1) 321 357 8740
1-408-341-0877
1-555-379-2296
(540) 433 - 9045
(415) 929 - 7305
818-491-2148
987-654-3210
949.497.5318
222.111.3333
206 566-5471
123 456-7890
510 357 8740
333 333 4444
(123) 456 7890
(234) 567 8901
949-497-5318
phonePhrases
unitRoom
addressLine
Gram
Ana
Text
phoneNumber.pat (pass 1)
\- [s]
_xNUM [s layer=(_suffix)]
@@
@PRE
<2,2> length(3)
<5,5> length(3)
<9,9> length(4)
@RULES
Ex: (415)_929_-_7305
_phoneNumber <-
_xWILD [min=1 max=1 s match=(_openPunct \()]
_xNUM [s layer=(_areaCode)]
_xWILD [min=1 max=1 s match=(_closePunct \))]
_xWHITE [star s]
_xNUM [s layer=(_prefix)]
_xWHITE [star s]
\- [s]
_xWHITE [star s]
_xNUM [s layer=(_suffix)]
@@
@PRE
<1,1> length(3)
<3,3> length(3)
<5,5> length(4)
@RULES
602
600
606

FIG. 12

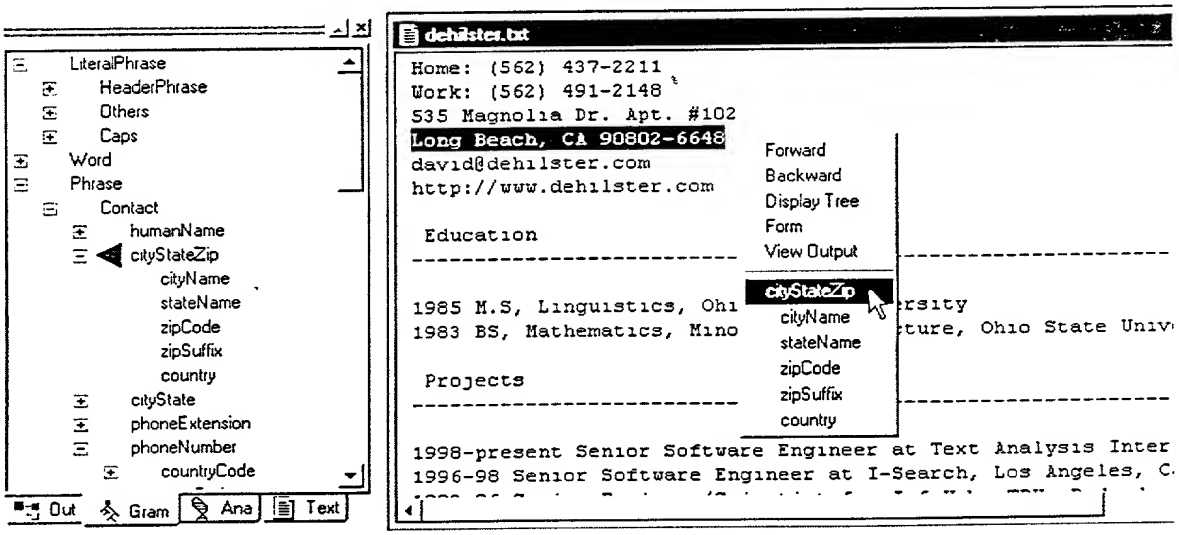

LiteralPhrase
HeaderPhrase
Others
Caps
Word
Phrase
Contact
humanName
cityStateZip
cityName
stateName
zipCode
zipSuffix
country
cityState
phoneExtension
phoneNumber
countryCode
Out
Gram
Ana
Text
dehilster.txt
Home: (562) 437-2211
Work: (562) 491-2148
535 Magnolia Dr. Apt. #102
Long Beach, CA 90802-6648
david@dehilster.com
http://www.dehilster.com
Education
1985 M.S, Linguistics, Ohi
1983 BS, Mathematics, Mino
Projects
1998-present Senior Software Engineer at Text Analysis Inter
1996-98 Senior Software Engineer at I-Search, Los Angeles, C.
Forward
Backward
Display Tree
Form
View Output
cityStateZip
cityName
stateName
zipCode
zipSuffix
country

FIG. 13

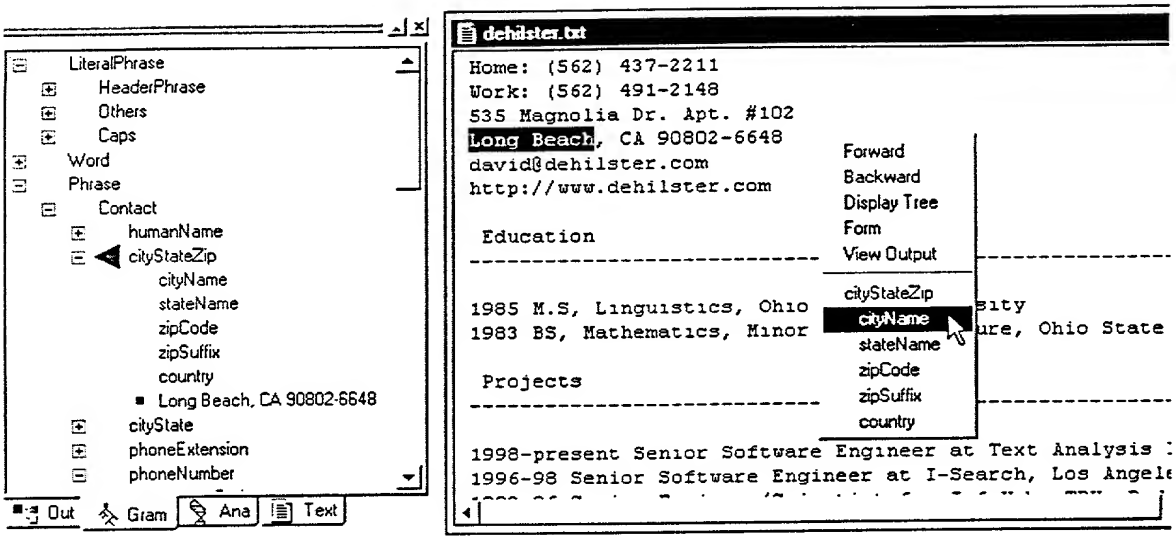

LiteralPhrase
HeaderPhrase
Others
Caps
Word
Phrase
Contact
humanName
cityStateZip
cityName
stateName
zipCode
zipSuffix
country
Long Beach, CA 90802-6648
cityState
phoneExtension
phoneNumber
Out
Gram
Ana
Text
dehilster.txt
Home: (562) 437-2211
Work: (562) 491-2148
535 Magnolia Dr. Apt. #102
Long Beach, CA 90802-6648
david@dehilster.com
http://www.dehilster.com
Education
1985 M.S, Linguistics, Ohio
1983 BS, Mathematics, Minor
Projects
1998-present Senior Software Engineer at Text Analysis
1996-98 Senior Software Engineer at I-Search, Los Angel
Forward
Backward
Display Tree
Form
View Output
cityStateZip
cityName
stateName
zipCode
zipSuffix
country

FIG. 14

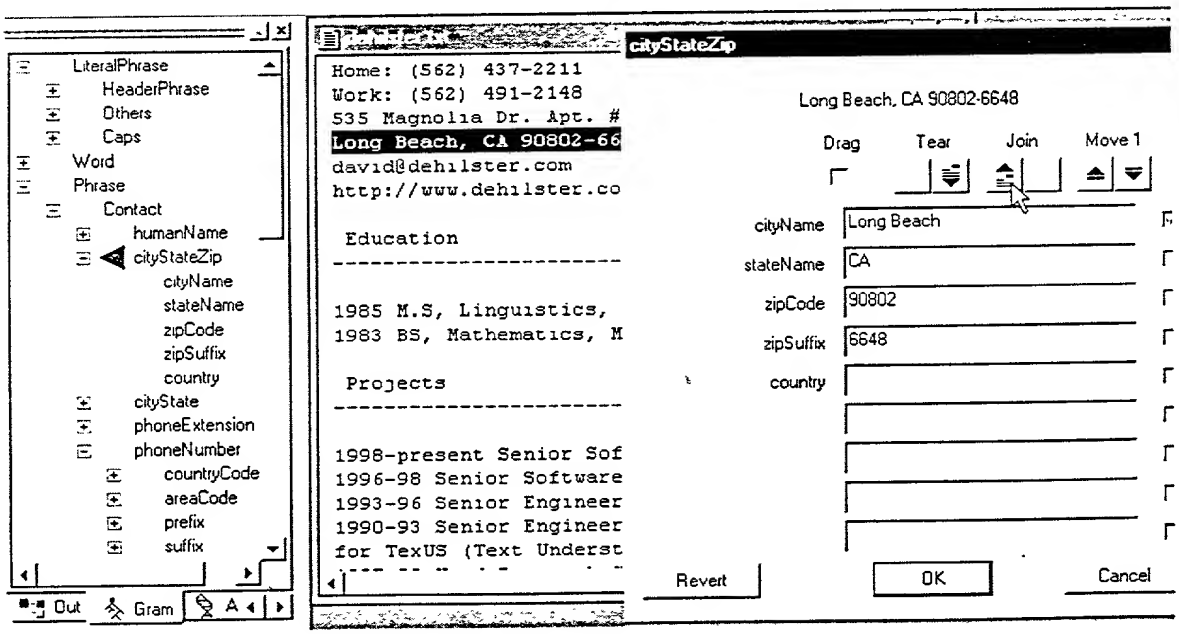

LiteralPhrase
HeaderPhrase
Others
Caps
Word
Phrase
Contact
humanName
cityStateZip
cityName
stateName
zipCode
zipSuffix
country
cityState
phoneExtension
phoneNumber
countryCode
areaCode
prefix
suffix
Out
Gram
A
Home: (562) 437-2211
Work: (562) 491-2148
535 Magnolia Dr. Apt. #
Long Beach, CA 90802-66
david@dehilster.com
http://www.dehilster.co
Education
1985 M.S, Linguistics,
1983 BS, Mathematics, M
Projects
1998-present Senior Sof
1996-98 Senior Software
1993-96 Senior Engineer
1990-93 Senior Engineer
for TexUS (Text Underst
cityStateZip
Long Beach, CA 90802-6648
Drag
Tear
Join
Move 1
cityName
Long Beach
stateName
CA
zipCode
90802
zipSuffix
6648
country
Revert
OK
Cancel

FIG. 15

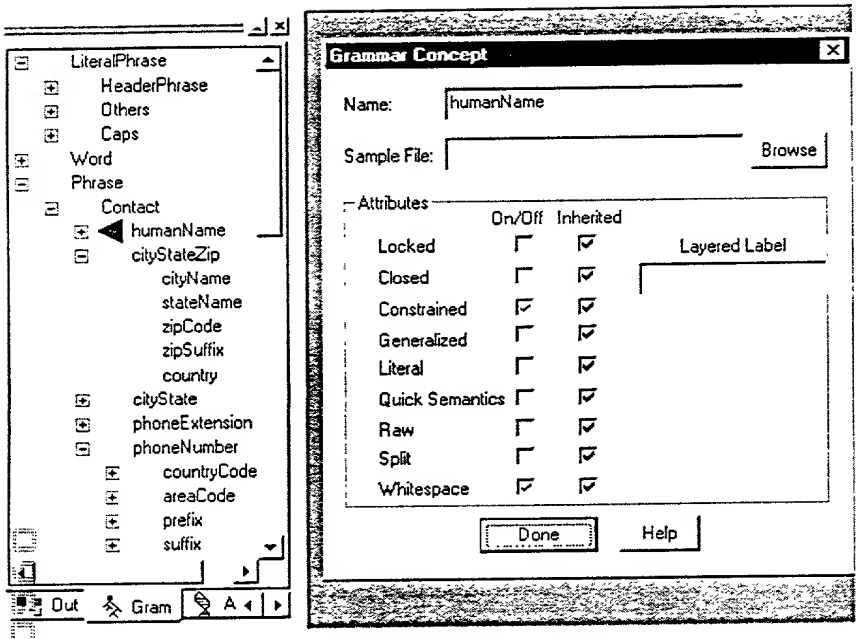


FIG. 16

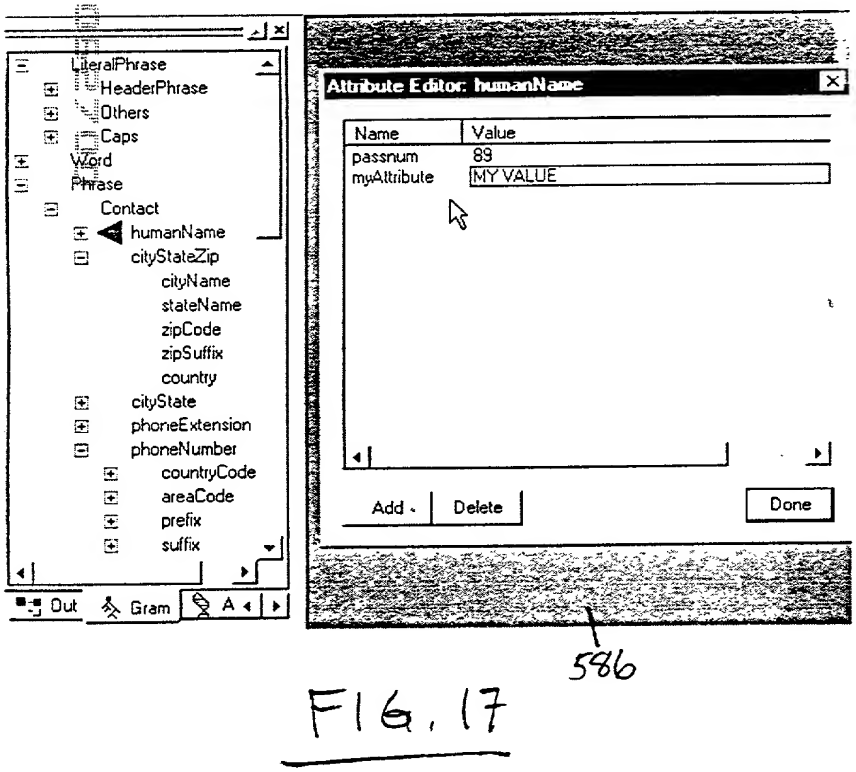


FIG. 17

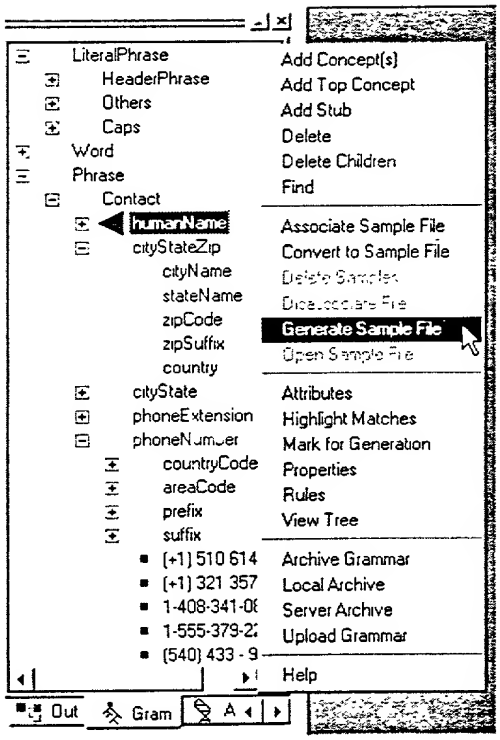


FIG. 18

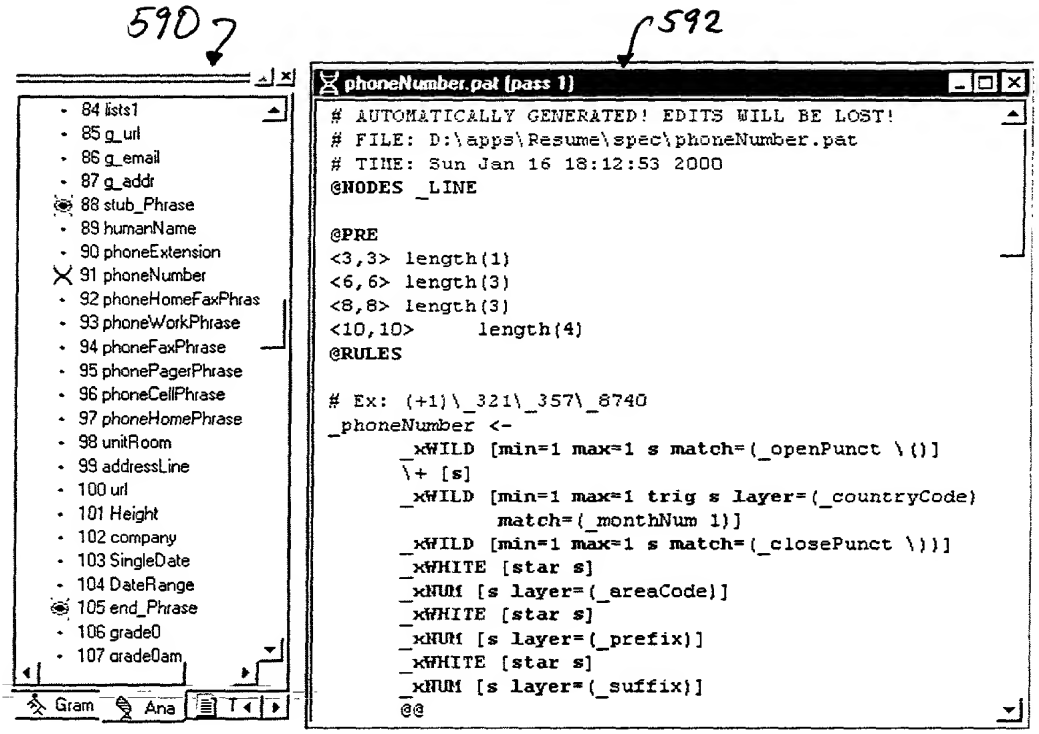

590
592
84 lists1
85 g_url
86 g_email
87 g_addr
88 stub_Phrase
89 humanName
90 phoneExtension
91 phoneNumber
92 phoneHomeFaxPhras
93 phoneWorkPhrase
94 phoneFaxPhrase
95 phonePagerPhrase
96 phoneCellPhrase
97 phoneHomePhrase
98 unitRoom
99 addressLine
100 url
101 Height
102 company
103 SingleDate
104 DateRange
105 end_Phrase
106 grade0
107 grade0am
Gram
Ana
phoneNumber.pat (pass 1)
AUTOMATICALLY GENERATED! EDITS WILL BE LOST!
FILE: D:\apps\Resume\spec\phoneNumber.pat
TIME: Sun Jan 16 18:12:53 2000
@NODES _LINE
@PRE
<3,3> length(1)
<6,6> length(3)
<8,8> length(3)
<10,10> length(4)
@RULES
Ex: (+1)_321_357_8740
_phoneNumber <-
_xWILD [min=1 max=1 s match=(_openPunct \()]
\+ [s]
_xWILD [min=1 max=1 trig s layer=(_countryCode)
match=(_monthNum 1)]
_xWILD [min=1 max=1 s match=(_closePunct \))]
_xWHITE [star s]
_xNUM [s layer=(_areaCode)]
_xWHITE [star s]
_xNUM [s layer=(_prefix)]
_xWHITE [star s]
_xNUM [s layer=(_suffix)]
@@

FIG. 19

DECLARATION FOR PATENT APPLICATION

As a below-named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below next to my name,

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled **AUTOMATED GENERATION OF TEXT ANALYSIS SYSTEMS**, the specification of which:

(check one)

☒ is attached hereto.

☐ was filed on _____ as Application Serial No. _____ and was amended on _____. (if applicable)

I hereby state that I have reviewed and understand the contents of the above-identified specification, including the claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose to the Patent Office all information known to me to be material to patentability as defined in 37 C.F.R. 1.56.

I hereby claim foreign priority benefits under Title 35, United States Code, §119 of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

Prior Foreign Application(s)			Priority Claimed	
			☐	☐
(Number)	(Country)	(Day/Month/Year Filed)	Yes	No

I hereby claim the benefit under Title 35, United States Code, §119(e) of any United States provisional application(s) listed below:

(Application Serial No.)	(Filing Date)	(Status) (patented, pending, abandoned)

I hereby claim the benefit under Title 35, United States Code, §120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, §112, I acknowledge the duty to disclose to the Patent Office all information known to me to be material to patentability as defined in 37 C.F.R. 1.56 which occurred between the filing date of the prior application and the national or PCT international filing date of this application:

(Application Serial No.)	(Filing Date)	(Status) (patented, pending, abandoned)

Direct all telephone calls to Aldo J. Test at (650) 494-8700.

Address all correspondence to:

FLEHR HOHBACH TEST
ALBRITTON & HERBERT LLP
Suite 3400, Four Embarcadero Center
San Francisco, California 94111

File No. A-68807/AJT/JWC

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Title 18, United States Code, §1001 and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Full name of first or sole inventor: Amnon Meyers

Inventor's signature: Amnon Meyers

Date: 6/19/00

Residence: Laguna Beach, California

Citizenship: USA

Post Office Address: 1261 Starlit Drive

Laguna Beach, California 92651

POWER OF ATTORNEY BY ASSIGNEE
AND EXCLUSION OF INVENTOR/S UNDER 37 C.F.R. §1.32
AND STATEMENT UNDER 37 C.F.R. § 3.73(b)

To the Commissioner of Patents and Trademarks:

The undersigned assignee of the entire interest in application for letters patent submitted herewith entitled AUTOMATED GENERATION OF TEXT ANALYSIS SYSTEMS, filed herewith and having the named inventor(s):

Avi Meyers

hereby appoints the following attorneys to prosecute this application and to transact all business in the Patent and Trademark Office connected therewith; said appointment to be to the exclusion of the inventor(s) and his (their) attorney(s) in accordance with the provisions of 37 C.F.R. 1.32: Harold C. Hohbach, Reg. No. 17,757; Aldo J. Test, Reg. No. 18,048; Donald N. MacIntosh, Reg. No. 20,316; Edward S. Wright, Reg. No. 24,903; David J. Brezner, Reg. No. 24,774; Richard E. Backus, Reg. No. 22,701; James A. Sheridan, Reg. No. 25,435; Robert B. Chickering, Reg. No. 24,286; Richard F. Trecartin, Reg. No. 31,801; Steven F. Caserza, Reg. No. 29,780; Michael A. Kaufman, Reg. No. 32,988; Edward N. Bachand, Reg. No. 37,085; R. Michael Ananian, Reg. No. 35,050; Stephen M. Kneuer, Reg. No. 38,208; Robin M. Silva, Reg. No. 38,304; David C. Ashby, Reg. No. 36,432; Maria S. Swiatek, Reg. No. 37,244, Dolly A. Vance, Reg. No. 39,054, Julian Caplan, Reg. No. 14,785, Brian G. Hart, Reg. No. 44,421; Steven M. Freeland, Reg. No. 42,555; William E. Nuttle; Reg. No. 42,943; and Victor E. Johnson, Reg. No. 41,546, provided that if any one of said attorneys ceases being affiliated with the law firm of Flehr Hohbach Test Albritton & Herbert LLP as partner, employee or of counsel, such attorney's appointment as attorney and all powers derived therefrom shall terminate on the date such attorney ceases being so affiliated.

Direct all telephone calls to Aldo J. Test at (650) 494-8700. Address all correspondence to:

FLEHR HOHBACH TEST
ALBRITTON & HERBERT LLP
Suite 3400, Four Embarcadero Center
San Francisco, California 94111-4187

Text Analysis International Inc., a Corporation of the State of California, certifies that it is the assignee of the entire right, title and interest in the patent application identified above by virtue of an assignment from the inventor of the patent application identified above. A copy of the executed, unrecorded assignment is attached hereto. The undersigned (whose title is supplied below) is empowered to act on behalf of the assignee.

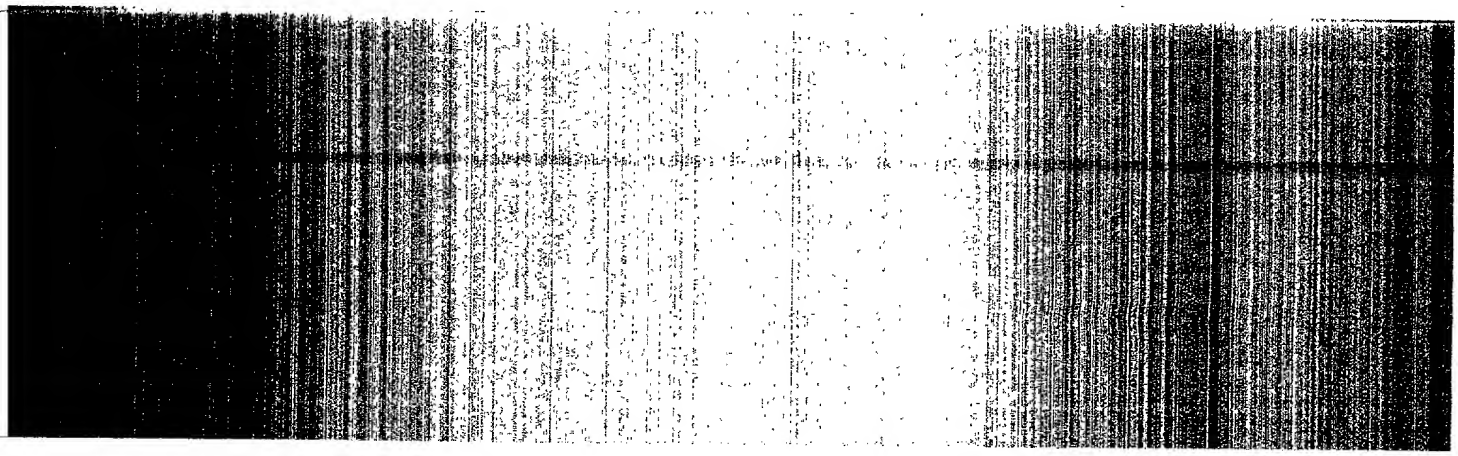
TEXT ANALYSIS INTERNATIONAL, INC.

Date: 6-26-00

By: _____

Avi Meyers, CEO
Text Analysis International, Inc.
1604 Mariani Drive
Sunnyvale, California 94087

A-68807/AJT/JWC **1012246**



United States Patent & Trademark Office

Office of Initial Patent Examination -- Scanning Division

Application deficiencies were found during scanning:

☐ Page(s)_____ of _____ were not present for scanning. (Document title)

☐ Page(s)_____ of _____ were not present for scanning. (Document title)

☒ Scanned copy is best available. of Declarations, Small entity.